Simulación de un Robot de Dos Grados de Libertad Usando "Hardware-in-the-Loop" con base en Instrumentación Programable

Fermín Hugo Ramírez Leyva

Universidad Tecnológica de la Mixteca, Carretera a Acatlima km 2.5 Huajuapan de León, Oaxaca, C.P. 69000, Tel. (01953) 5320399 ext. (555)

hugo@mixteco.utm.mx

Luis Alberto Pérez Gaspar

Universidad Tecnológica de la Mixteca, Carretera a Acatlima km 2.5 Huajuapan de León, Oaxaca, C.P. 69000, Tel. (01953) 5320399 ext. (555)

Felipe Santiago Espinosa

Universidad Tecnológica de la Mixteca, Carretera a Acatlima km 2.5 Huajuapan de León, Oaxaca, C.P. 69000, Tel. (01953) 5320399 ext. (555)

fsantiag@mixteco.utm.mx

Resumen

En este trabajo se muestra la forma en que se hizo un sistema, con base en instrumentación programable (LabVIEW), para emular el modelo dinámico de un robot antropomórfico de dos grados de libertad con "*Hardware-in-the-loop*". El programa de control está basado en una computadora personal, que resuelve las ecuaciones diferenciales en tiempo real, convierte los valores numéricos de las posiciones y velocidades y toma el par de entrada con señales de voltajes con tarjetas de adquisición de datos por USB. La interfaz gráfica muestra el historial de la evolución de las variables del sistema y la representación gráfica de los movimientos del robot.

Palabras Claves: Simulación de control de robots, Hardware-in-the-loop, LabVIEW

1. Introducción

Cuando se trabaja con sistemas de control, el primer paso es obtener el modelo dinámico de la planta, que puede ser lineal o no lineal. Posteriormente se simula y prueban las diferentes estrategias de control, finalmente se pasa a la implementación. Sin embargo en ocasiones, por cuestiones económicas o disponibilidad del equipo, no es posible realizar esta última fase.

La simulación basada únicamente en software como Matlab/Simulink o Simnon no tiene capacidad de reproducir en forma más aproximada las condiciones reales de operación, tanto del controlador como de la planta. Una forma de mejorarlo es usando “Hardware-in-the-loop” (HIL). Con esta tecnología se permite la interacción completa y correcta de un dispositivo real (el regulador o controlador) con una simulada (la planta). Lo que aumenta el realismo de la simulación y proporciona acceso a las características de Hardware, actualmente no disponibles en los modelos de simulación basados sólo en software. Por lo tanto, reduce los riesgos de descubrir un error en la última etapa de la prueba en campo [1].

Los sistemas más usados para implementar HIL son dSpace y LabVIEW [2]. Los basados en dSpace usan Matlab/Simulink, lo que facilita el desarrollo de modelos. El LabVIEW usa su mismo entorno de desarrollo para hacer estas aplicaciones. La ventaja principal de usar estos sistemas es que facilita el diseño y prueba de modelos, pero su principal desventaja es el costo, ya que el Hardware requerido es caro y su precio es inversamente proporcional al período de muestreo.

El área de robótica ha tenido un gran desarrollo y aceptación en los últimos años. En particular el control de robots manipuladores representa un reto matemático y de control importante, ya que son sistemas no lineales de múltiples entradas y salidas. Las simulaciones permiten entender cómo funcionan los controladores para cumplir con el objetivo de control. Para probar físicamente las estrategias de control se necesita tener acceso a un robot de arquitectura abierta. Sin embargo en el país pocas instituciones cuentan con este tipo de infraestructura.

A nivel educativo el diseñar el modelo de un robot manipulador usando HIL, da la posibilidad de que se experimenten las técnicas de control aprendidas en los cursos de robótica, usando equipo que se tiene en la mayoría de los laboratorios de electrónica. También tiene la ventaja de que se pude reproducir fácilmente y no es necesario construir o comprar un robot de arquitectura abierta.

Algunos trabajos que se han encontrado en la literatura que usan HIL son: Marco Morandini presenta la simulación con HIL de un robot de 6 grados de libertad (GDL) y usa un clúster de máquinas para la simulación, usando software libre para la implementación de los algoritmos de tiempo real [3]. Xiumin Diao usa HIL para determinar el espacio de trabajo de un robot paralelo de 6GLD impulsado por cable para aplicaciones de micro gravedad [4]. Paolo Boscariol presenta la simulación en HIL de un péndulo flexible, el cual resuelve en tiempo real el modelo no lineal por elemento finito y hace pruebas con el controlador y la planta real [5].

Una aplicación de HIL con equipo de bajo costo la presenta Bin Lu que realiza la simulación de convertidores CD-CD, los cuales tienen restricciones importantes en la respuesta de tiempo real [1]. Panayiotis hace la simulación de un péndulo invertido controlado con redes neuronales, para ello usa 2 computadoras personales y Matlab/Simulink [6].

De la investigación bibliográfica que se hizo, la mayoría de trabajos relacionados con HIL usan Matlab/Simulink como entrono de desarrollo, y Hardware caro, como son módulos PXI o VXI y sistemas operativos de tiempo real. En este trabajo se usa LabVIEW para implementar el modelo de un robot de dos grados de libertad con HIL y Hardware de bajo costo. Con tarjetas de adquisición de datos (TAD) se adquieren la señal de entrada y se generan los voltajes de salida. En las siguientes secciones se explica el modelo matemático del robot, la forma en que se configuró el HIL y los resultados obtenidos.

2. Modelo del robot manipulador de dos GDL

El modelo matemático de un robot de *n* grados de libertad está definido por la ecuación (1). Donde $M(q)\varepsilon R^{n\times n}$ es la matriz de Inercia, $C(q,\dot{q})\varepsilon R^{n\times n}$ es la matriz de Coriolis y

fuerza centrífuga, $g(q)\varepsilon R^n$ el par gravitacional, $f(\dot{q})\varepsilon R^n$ el par de fricción y $\tau\varepsilon R^n$ el par aplicado a cada una de las articulaciones. Los términos $\ddot{q},\ \dot{q}\ y\ q\ \varepsilon R^n$ representan a los vectores de aceleración, velocidad y posición [7].

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(\dot{q}) = \tau \quad (1)$$

Para un robot antropomórfico de dos grados de libertad *n*=2, su modelo matemático se compone de dos ecuaciones diferenciales no lineales. En forma matricial la ecuación (1) se expresa como se muestra en la ecuación (2), donde $q = [q_1 \quad q_2]^T$, $\dot{q} = [\dot{q}_1 \quad \dot{q}_2]^T$ y $\ddot{q} = [\ddot{q}_1 \quad \ddot{q}_2]^T$ son los vectores de posición, velocidad y aceleración, y los subíndices indican la articulación. Las matrices de Inercia y Coriolis constan de 4 términos; $\tau = [\tau_1 \quad \tau_2]^T$, $g = [g_1 \quad g_2]^T$ y $f = [fric_1 \quad fric_2]^T$ son el par de entrada, el par gravitacional y de fricción, cada uno tiene 2 elementos.

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}\ddot{q} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}\dot{q} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} + \begin{bmatrix} fric_1 \\ fric_2 \end{bmatrix} = \tau \quad (2)$$

La ecuación (2) se debe poner en forma de un sistema de ecuaciones diferenciales de segundo grado, para lo cual se despeja el vector de aceleración y las ecuaciones resultantes son la (3) y (4). En ellas $M^{-1}(q)$ es la matriz inversa de la inercia.

$$\ddot{q} = M^{-1}(q)[\tau - C(q,\dot{q})\dot{q} - g(q) - f(\dot{q})] \quad (3)$$

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = \frac{1}{M_{11}M_{22} - M_{21}M_{12}} \begin{pmatrix} M_{22} & -M_{12} \\ -M_{21} & M_{11} \end{pmatrix} \left(\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} - \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} - \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} - \begin{bmatrix} fric_1 \\ fric_2 \end{bmatrix} \right) \quad (4)$$

Para simular el sistema de la ecuación (4) se requiere tener los componentes del modelo dinámico, los cuales se pude deducir usando el método de Euler-Lagrange. En la referencia [8] se describen los pasos para obtener cada uno de los términos de la

ecuación (4), los cuales corresponde a lo especificados en la ecuación (5). Como se observa, los componentes del modelo dinámico son función de las masas, longitud, centro de masa de los eslabones, las posiciones y velocidades. La definición de cada término de la ecuación (4) está en la tabla 1. Para la fricción sólo se toma en cuenta la fricción viscosa.

$$M_{11} = m_1 l_{c1}^2 + m_2[l_1^2 + l_{c2}^2 + 2l_1 l_{c2}\cos(q_2)] + I_1 + I_2$$

$$M_{12} = m_2[l_{c2}^2 + l_1 l_{c2}\cos(q_2) + I_2]$$

$$M_{21} = m_2[l_{c2}^2 + l_1 l_{c2}\cos(q_2) + I_2]$$

$$M_{22} = m_2 l_{c2}^2 + I_2$$

$$C_{11} = -m_2 l_1 l_{c2}\sin(q_2)\,\dot{q}_2 \qquad (5)$$

$$C_{12} = -m_2 l_1 l_{c2}\sin(q_2)\,[\dot{q}_1 + \dot{q}_2]$$

$$C_{21} = m_2 l_1 l_{c2}\sin(q_2)\,\dot{q}_1$$

$$C_{22} = 0$$

$$g_1 = [m_1 l_{c1} + m_2 l_1]gsin(q_1) + m_2 l_{c2} gsin(q_1 + q_2)$$

$$g_2 = m_2 l_{c2} gsin(q_1 + q_2)$$

$$fric_1 = b_1\dot{q}_1$$

$$fric_2 = b_2\dot{q}_2$$

Para realizar la simulación del robot es necesario contar con el valor de los parámetros. Se usó el modelo del robot de arquitectura abierta de la Benemérita Universidad Autónoma de Puebla (BUAP). Este robot consta de tres grados de libertad, sus eslabones están hechos de aluminio 6061 y los actuadores son motores de transmisión directa (ver

Fig. 1). Los 2 últimos eslabones se pueden considerar como un robot de dos GDL. En la tabla 1 se presenta un listado de los parámetros físicos del robot [8].

Tabla 1. Elementos del modelo del robot de dos GDL de la BUAP.

Parámetro	Símbolo	Valor	Unidad
Longitud de eslabón 1	l_1	0.45	m
Longitud de eslabón 2	l_2	0.45	m
Masa del eslabón 1	m_1	23.902	kg
Masa del eslabón 2	m_2	3.880	kg
Centro de masa del eslabón 1	l_{c1}	0.091	m
Centro de masa del eslabón 2	l_{c2}	0.048	m
Momento de Inercia 1	I_1	1.266	kg m^2
Momento de Inercia 2	I_2	0.093	kg m^2
Coeficiente de viscosidad 1	b_1	2.288	Nm s
Coeficiente de viscosidad 2	b_2	0.175	Nm s
Coeficiente de Coulomb 1	f_{c1}	7.17	Nm
Coeficiente de Coulomb 2	f_{c2}	1.734	Nm
Aceleración de la gravedad	g	9.81	m/s^2
Par máximo de la articulación 1	τ_1	150	Nm
Par máximo de la articulación 2	τ_2	15	Nm

Fig. 1. Robot de transmisión directa de la BUAP

3. Implementación del robot con HIL

La implementación del HIL del robot de dos GDL se hizo con base en LabVIEW, una computadora personal y dos tarjetas de adquisición de datos (TAD). La versión de LabVIEW usada es la 2010 con sistema operativo Windows 7®. La digitalización de las señales de entrada y salida analógicas se hace con las dos TAD, debido a que cada una cuenta con dos salidas analógicas y se requieren 4. El procedimiento usado para hacer el HIL es:

- Adquirir las señales de voltaje proporcional al par de cada articulación.
- Escalarlas para tener el par equivalente.
- Resolver numéricamente las ecuaciones diferenciales.
- Escalar las salidas de posición y velocidad para convertirlas en voltaje.
- Esperar a que concluya el periodo de muestreo para repetir el proceso.

Las TAD usadas son de la firma National Instruments modelo NI USB 6008 y la USB1208FS de la firma Measurement Computing. La primera tiene 8 entradas analógicas referenciadas, voltaje de entrada de $\pm$10V, frecuencia de muestreo de 10kS/s, 2 salidas analógicas, con convertidores analógico digital y digital analógico de 12bits [9]. Con esta TAD se leen los voltajes del par y las salidas analógicas correspondientes a las posiciones de las articulaciones 1 y 2. La USB 1208FS tiene 8 señales de entrada en el rango de $\pm$10V, 2 salidas analógicas de 0 a 4.1V y frecuencia de muestreo de 50kS/s [10]. Con esta última se generan los voltajes proporcionales a las velocidades de las articulaciones 1 y 2. En la Fig. 2 se muestra el diagrama bloques de la conexión de los componentes del sistema.

Las ecuaciones diferenciales (4) se programan con métodos numéricos, se utilizó el algoritmo de Euler. Este método resuelve ecuaciones diferenciales de primer grado; por lo cual, cada ecuación del modelo del robot se debe dividir en dos ecuaciones de este orden, mediante su representación en variables de estado. El procedimiento para hacerlo se describe a continuación.

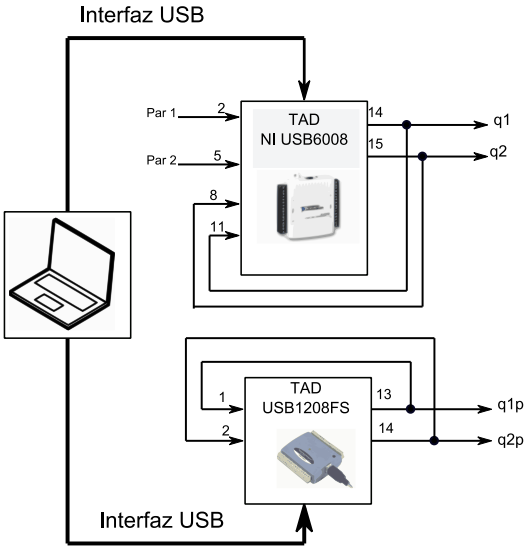

Fig. 2. Diagrama a bloques del sistema para implementar el HIL

Una ecuación diferencial (6) es de segundo grado y es función de x, $\dot{x}$ y la entrada *u*.

$$\ddot{x} = r(x, \dot{x}, u) \tag{6}$$

Su representación en variables de estado ($\boldsymbol{X} = [x_1 \quad x_2]^T$), está dada en (7), la cual se descompone en 2 ecuaciones diferenciales de primer grado.

$$\begin{aligned} x_1 &= x \\ \dot{x_1} &= x_2 \\ \dot{x_2} &= r(x, \dot{x}, u) \end{aligned} \tag{7}$$

Para resolver numéricamente la ecuación (7) con el algoritmo de Euler se cambia a tiempo discreto. La ecuación (8) es la representación en tiempo discreto de la ecuación (7), donde $\mathrm{x}_1(k)$, $\mathrm{x}_2(\mathrm{k})$, $\mathrm{x}_1(\mathrm{k}-1)$ y $\mathrm{x}_2(\mathrm{k}-1)$ contienen el valor actual (k) y anterior (k-1) de las variables de estado (x_1 y x_2), $u(k)$ es la entrada discretizada y *h* es el período de muestreo. En el robot de dos GDL las variables de estado son las posiciones y velocidades de cada articulación.

$$x_1(k) = x_1(k-1) + h * x_2(k-1) \quad (8)$$

$$x_2(k) = x_2(k-1) + h * r\big(x_1(k-1), x_2(k-1), u(k)\big)$$

El modelo dinámico del robot de dos GDL se programó en LabVIEW. En este programa se usaron dos nodos de fórmula, que usan la sintaxis de C estándar, para facilitar su codificación. En la Fig. 3 se muestra el código de la función que resuelve la ecuación diferencial. Las entradas son el estado anterior, que viene en un Clúster, el par y período de muestreo (*h*), la salida es el estado actual. El primer nodo de fórmula evalúa la ecuación (5). El segundo nodo de fórmula resuelve las 2 ecuaciones diferenciales de tiempo discreto con el algoritmo de la ecuación (8). Esta función se ejecuta cada *h* segundos y su salida es el estado actual que *h* segundo después entra como estado anterior.

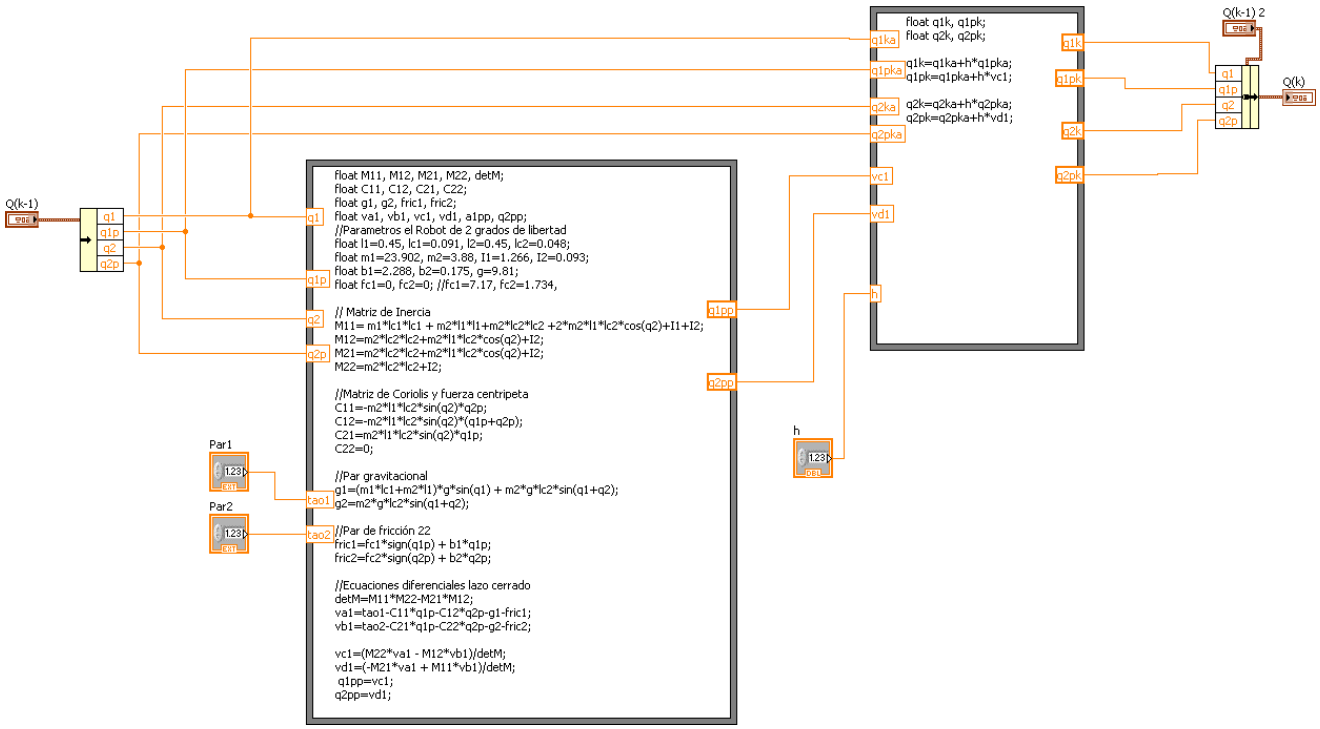

Fig. 3. Código de las funciones del robot que resuelve la ecuación diferencial

En la Fig. 4 se muestra el Ícono de la función cuyas entradas son el Clúster del estado anterior, el par 1 y 2 y el período de muestreo y su salida el estado actual. Se usó un icono predefinido, pero el código que tiene programado es el de la Fig. 3.

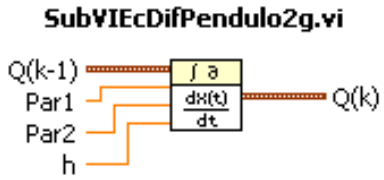

Fig. 4. Ícono de la función que resuelve la ecuación diferencial

Las entradas y salidas de la ecuación diferencial calculadas numéricamente pueden tomar valores muy grandes. En esta aplicación se convierten a voltajes con una TAD, por lo que es necesario escalarlo al rango de valores que pueden manejar. Para el par de entrada de la articulación 1 y 2 (τ_1 y τ_2) el valor máximo del par es de $\pm$ 150Nm y $\pm$15Nm respectivamente. Éstos se escalan al rango de $\pm$10V, que es el nivel de las entradas analógicas de la TAD. La ecuación (9) hace el escalamiento del par 1 y 2, donde Vin_0 y Vin_1 son los voltajes medidos en las entradas analógicas 0 y 1 de la TAD.

$$\tau_1 = 15 * Vin_0 \qquad (9)$$

$$\tau_2 = 1.5 * Vin_1$$

Las salidas de posición y velocidad se acotan a voltajes en el rango de 0 a 4V. Las variables de posición y velocidad se consideran que están en el rango $\pm\pi$. Para manejar voltajes positivos y negativos se usa un offset de 2V. La función de transferencia para cualquier posición o velocidad es la (10), donde $V(q_n)$ es el voltaje de salida y q_n la posición o velocidad a convertir.

$$V(q_n) = 2V + \frac{2}{\pi} q_n \qquad (10)$$

Las pruebas de funcionamiento del sistema se hicieron a lazo abierto. Con un generador de funciones se le aplica voltaje proporcional al par de las 2 entradas. Las salidas se miden con osciloscopio y con la misma tarjeta de adquisición de datos, para guardar el historial de su comportamiento. En la Fig. 5 se muestra el panel frontal del programa que hace el HIL y en la Fig. 6 su diagrama a bloques.

El panel frontal de la aplicación consta de 3 pestañas (Configuración, Qp y Q). En Configuración se definen las condiciones iníciales de las posiciones y velocidades (por default son cero), y muestra el tiempo promedio que tarda en ejecutar cada iteración. En la pestaña Qp se despliega la gráfica de las velocidades y el par de entrada. En la pestaña Q (mostrada en la Fig. 5), se muestra la respuesta de las posiciones de la articulación 1 y 2 (q_1 y q_2), y además de la gráfica de un robot de dos GDL, la cual se mueve en función de las salidas de posición del modelo dinámico.

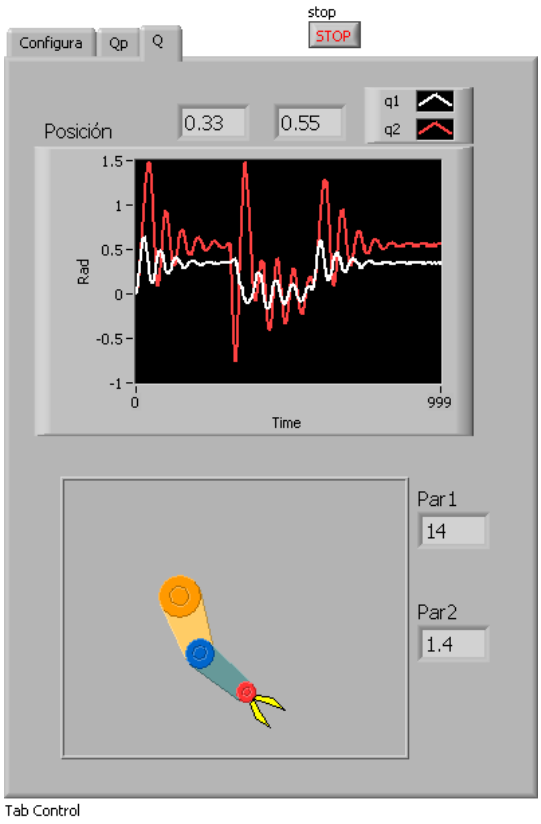

Fig. 5. Panel frontal del programa de HIL

La programación del HIL en LabVIEW se hizo modular y se dividió en 4 funciones (llamadas subVI). Las cuales son:

- Lectura de entradas analógicas (pares, posiciones y velocidades).
- Escritura de salidas analógicas (posiciones y velocidades).
- Solución de ecuaciones diferenciales.
- Graficación de movimiento del robot.

Experimentalmente, se encontró que capturar una o varias señales analógicas y enviarlas a la salida con la TAD, se tarda entre 10ms y 30ms. Este tiempo es independiente de la frecuencia de muestreo con el que se configure la TAD (siempre y cuando fuera mayor a 1kS/s), pero dependía del tipo de computadora usada. El tiempo que tarda en calcular la ecuación diferencial es muy corto y no es significativo en comparación con el manejo de la TAD.

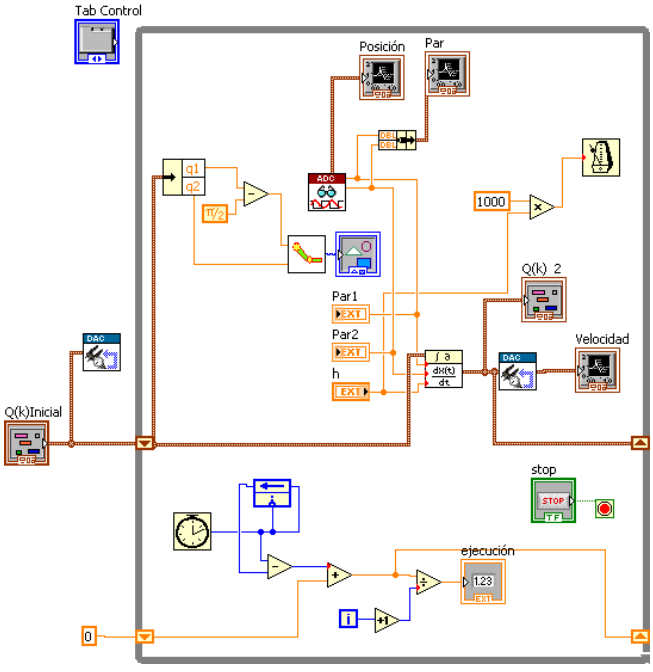

Fig. 6. Diagrama a bloques del programa de HIL

En el equipo que se hicieron las pruebas, el período de muestreo a partir del cual cada iteración se ejecutaba con un tiempo constante fue h=30ms. Para verificar el funcionamiento del sistema se comparó la respuesta medida contra la que obtenida del simulador matemático SIMNON, con h=30ms. En la Fig. 7 se muestra la comparación de las respuestas para las posiciones medidas y simuladas del eslabón 1 y 2. El algoritmo del simulador fue el Runge-Kutta-Fehlberg 4/5. En el HIL los voltajes de entrada fueron de 1V, que corresponde a un par de 14.05 y 1.40 Nm, en los eslabones 1 y 2 respectivamente.

Analizado la respuesta de la posición de la articulación 1 y 2 (q_1 y q_2), se observa que ambas inician en 0 rad, tienen una respuesta subamortiguada similar. El sobretiro es mayor con HIL que con el simulador. Finalmente ambas convergen al mismo punto, que es 0.34rad (19.48º) y de 0.54rad (30.93º) para la posición 1 y 2. Las simulaciones con HIL tienen ruido de cuantización cuando las salidas presentan cambios pequeños.

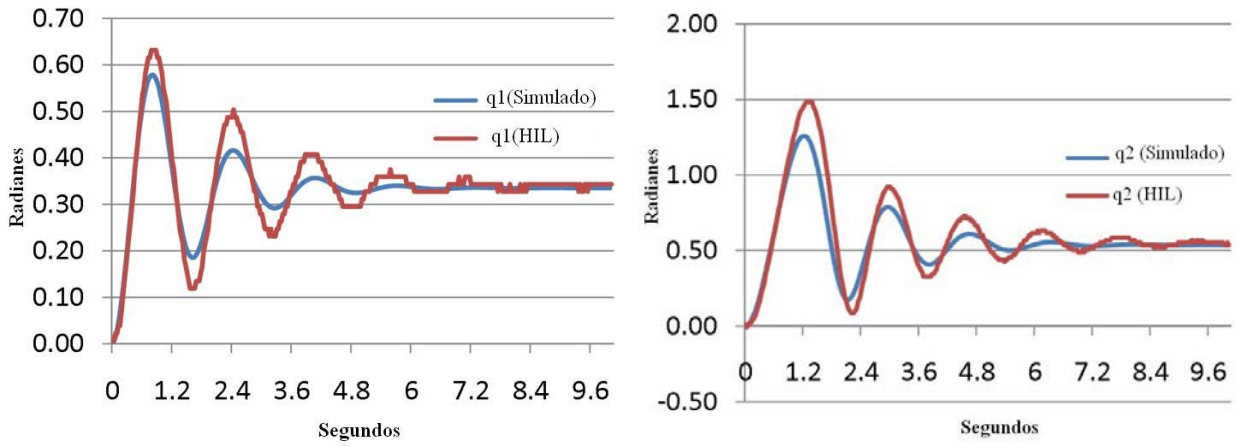

Fig. 7. Respuesta simulada y con HIL de las posiciones 1 y 2

4. Conclusiones

En este trabajo se mostró la forma en que se implementó el HIL de un robot de dos grados de libertad, usando LabVIEW como ambiente de programación, una computadora personal y tarjetas de adquisición de datos. El tiempo de muestreo que se consiguió fue de 30ms, el cual fue suficiente para ejecutar el modelo del sistema. Es importante trabajar en la reducción de este tiempo ya que a medida que sea menor se tendrá una mejor aproximación del sistema real con el HIL. Con este Hardware no es posible, por lo que se va a trabajar con tarjetas de adquisición de datos más rápidas, o se buscará emplear otros sistemas de desarrollo como son Controladores Digitales de Señales, Microcontroladores con unidad de punto flotante o FPGA's.

La principal contribución que se tiene fue la demostración que con Hardware de bajo costo es posible simular sistemas mecánicos complejos, como es el caso de un robot de dos grados de libertad, ya que las ecuaciones diferenciales se resuelven en tiempo real. LabVIEW tiene métodos programados para resolver ecuaciones diferenciales, pero en esta aplicación se tuvo que programar todo para resolver el modelo en tiempo real. Otra ventaja de usar LabVIEW es que permite mostrar en forma gráfica cómo se mueve un mecanismo, de forma similar a como lo hace el robot real. Por lo cual en trabajos futuros se trabajará en la de la interfaz gráfica que simula el movimiento del robot, con el fin de que se parezca cada vez más al sistema real.

Una vez que se tiene implementado el sistema con HIL, el siguiente paso es la evaluación de los controladores de posición y movimiento que se ven en los cursos de robótica como son el PD con compensación de gravedad y PD+. La ventaja de emular en lugar de simular, es que se tienen que resolver los problemas que se presenta cuando se trabaja con el robot real, como son la interconexión y el ruido. Desde el punto de vista educativo este esquema es más enriquecedor a sólo quedarse a nivel simulación sin interactuar con Hardware.

6. Referencias

[1] L. Bin, X. Wu, F. Hernán, A. Monti. "A low-Cost Real-Time Hardware-in-the-Loop Testing Approach of Power Electronics Controls". IEE Transactions on Industrial Electronics. Volumen 54, Número 2. Abril 2007.

[2] Hardware-in-the-loop Simulation. http://www.embedded.com/design/prototyping-and-development/4024865/Hardware-in-the-Loop-Simulation. Abril 2014.

[3] M. Morandini, P. Masarati, and P. Mantegazza. "A Real-Time Hardware-In-The-Loop Simulator For Robotics Applications". Multibody Dynamics 2005, ECCOMAS Thematic Conference. Junio 2005.

[4] Xiumin Diao, Ou Ma. "Workspace Analysis of a 6-DOF Cable Robot for Hardware-in-the-Loop Dynamic Simulation". Proceedings of the 2006 IEEE/RSJ, International Conference on Intelligent Robots and Systems, Octubre 2006.

[5] P. Boscariol, A. Gasparetto, V. Zanotto, “A HIL Simulator of flexible-link Mechanisms”, Journal of intelligent & Robotic System. Volumen 64. Número, 3-4. Páginas 427-446. Diciembre 2011.

[6] P. Shiakolas, S. Van Schenck, D. Piyabongkarn, L. Frangeskou. "Magnetic Levitation Hardware-in-the-Loop and MATLAB-Based Experiments for Reinforcement of Neural Network Control Concepts". IEEE Transactions on Education. Volumen. 47. Número 1. Febrero 2004.

[7] F. H. Ramírez, F. Reyes, R. Salazar. “A New Family of Saturated Regulators for Robot Manipulators”; Proceedings of the Tenth IASTED International Conference; Mayo 26-28, Quebec City, Quebec, Canada CONTROL AND APPLICATIONS (CA 2008). Página 102-106. Mayo 2008.

[8] R. Kelly, V. Santibáñez. “Control de Movimiento de Robots Manipuladores”. 2003. Pearson Prentice Hall. España. Página 65-69.

[9] USER GUIDE AND SPECIFICATIONS NI USB-6008/6009. National Instruments Corporation. 2012

[10] USB-1208FS User's Guide. Measurement Computing. 2014

7. Autores

M. en C. Fermín Hugo Ramírez Leyva obtuvo su título de Maestría en Ciencias con especialidad en electrónica por el Instituto Nacional de Astrofísica Óptica y Electrónica. Actualmente es profesor investigador, adscrito al Instituto de Electrónica y Mecatrónica de la Universidad Tecnológica de la Mixteca desde 1999. Es estudiante de doctorado en Ingeniería Mecatrónica en la Universidad Popular Autónoma del Estado de Puebla.

Ing. Luis Alberto Pérez-Gaspar, es egresado del Tecnológico de Veracruz, especialidad en sistemas digitales. Actualmente es estudiante de maestría en Robótica en la Universidad Tecnológica de la Mixteca.

M. C. Felipe Santiago Espinosa es Maestro en Ciencias con especialidad en Electrónica por parte del INAOE, incorporado al Instituto de Electrónica y Mecatrónica de la Universidad Tecnológica de la Mixteca, en donde es Profesor-Investigador desde 1998. En el año de 2012 publicó su libro titulado “Los Microcontroladores AVR de ATMEL”. Actualmente está cursando el Doctorado en Robótica en la misma institución.